Full Stack Web Development Masterclass

Mastering React.js & Django (Python)

A Project-Based, 16-Week Journey to Become a Market-Ready Developer

---

Lurnxo Vision & Pedagogy

The Lurnxo vision is to create developers, not just coders. Our curriculum is designed around the philosophy of **The Modern Developer Triangle**: *Foundations, Frameworks, and Flow*. We prioritize not only mastering the core languages and modern frameworks (React and Django), but also integrating essential development operations (DevOps) and productivity tools (AI for Coding) from day one.

- **Project-First Approach:** Every module culminates in a practical, deployable mini-project.
- **AI-Enhanced Productivity:** Learn to leverage AI tools (e.g., code completion, debugging, documentation) as a development partner to increase efficiency by up to 40%.
- **Deployment Ready:** We focus on the complete development lifecycle, ensuring you know how to build, test, secure, and deploy applications on cloud platforms.
- **Portfolio as Your Resume:** By the end, you will have a robust, deployed portfolio to showcase to employers.

Course Overview & Prerequisites

2.1 Target Outcome. Graduates will be proficient in building single-page applications (SPAs) with a RESTful API backend, capable of full CRUD (Create, Read, Update, Delete) operations, user authentication, and deployment.

2.2 Duration. 16 Weeks (Approx. 4 Months)

2.3 Tools & Technologies. **Frontend:** HTML5, CSS3, JavaScript ES6+, React, Tailwind/Bootstrap.
Backend: Python, Django, SQL, PostgreSQL.

Module 1: Frontend Immersion (Weeks 1-9)

3.1 Week 1-3: HTML & CSS Foundations.

- **HTML5:** Structure, Semantic Tags, Forms, Accessibility (ARIA).
- **CSS3:** Selectors, Flexbox, Grid Layout (Modern Layouts).
- **Responsive Design:** Media Queries, Viewport, Mobile-First Approach.
- **Styling Frameworks:** Introduction to a utility-first approach (Tailwind or Bootstrap).
- **Project 1:** Design and build a fully responsive Static Business Landing Page.

3.2 Week 4-6: Core JavaScript (ES6+).

- **Fundamentals:** Data Types, Control Flow, Functions, Scope (Closure).
- **Modern JS (ES6+):** Arrow Functions, Promises, Async/Await, Destructuring.
- **DOM Manipulation:** Interacting with the webpage, event handling, data fetching (Fetch API / Axios).
- **Object-Oriented JS:** Classes, Inheritance, Prototypes.
- **Project 2:** Build a functional To-Do List application with local data persistence.

3.3 Week 7-9: React.js Mastery.

- **Core Concepts:** Component Structure, JSX, Props, State Management (useState, useEffect).
- **React Hooks:** Deep dive into ‘useEffect‘, ‘useContext‘, ‘useReducer‘, and custom hooks.
- **Routing:** Client-side routing for Single Page Applications (SPAs).
- **State Management:** Introduction to advanced state patterns (e.g., Context API or Zustand).
- **Project 3:** Create a multi-page Movie or Recipe Explorer app using a public API.

Module 2: Backend Immersion (Weeks 10-14)

4.1 Week 10-11: Python Programming Core.

- **Python Syntax:** Variables, Loops, Conditionals, Data Structures (Lists, Dictionaries, Sets).
- **OOP in Python:** Classes, Methods, Inheritance.
- **Functional Concepts:** List Comprehensions, Lambda Functions, Generators.
- **Best Practices:** Virtual Environments, PEP 8 Style Guide, Unit Testing.
- **Project 4:** Build a Command Line Interface (CLI) application using Python.

4.2 Week 12-13: Django Fundamentals & APIs.

- **Django Setup:** Project and App structure, Settings, URL Routing.
- **Models & ORM:** Defining database structure, Migrations, Querying the database.
- **Django REST Framework (DRF):** Building Serializers, Views (Function-based and Class-based APIs).
- **Authentication:** Token-based authentication for React integration.
- **Project 5:** Create a robust RESTful API for a Blog application.

4.3 Week 14: SQL & PostgreSQL Database.

- **SQL Basics:** SELECT, INSERT, UPDATE, DELETE statements, Joins.
- **PostgreSQL:** Setup and configuration, advanced data types (JSON/Array).
- **Database Optimization:** Indexing, Query optimization, Relational Integrity.
- **Integration:** Connecting Django ORM to the PostgreSQL database for production-ready data storage.

Module 3: Integration & Career Readiness (Weeks 15-16)

5.1 Week 15: Full Stack Integration & Deployment.

- **Integration:** Connecting the React Frontend to the Django Backend API (CORS policies, API consumption).
- **Git & GitHub:** Version Control fundamentals, Branching, Merging, Pull Requests.
- **Deployment Strategy:** Setting up a Production environment.
- **Hosting:** Deploying the Django API (e.g., using Render/Heroku) and the React Frontend (e.g., using Netlify/Vercel).
- **Project 6 (Capstone Start):** Initiate the final Capstone Project.

5.2 Week 16: Advanced Tools & Portfolio.

- **AI for Coding:** Learning Prompt Engineering for development, using AI tools for refactoring, test generation, and documentation (e.g., GitHub Copilot, Claude).
- **Portfolio Making:** Strategies for structuring and presenting the final portfolio (The "Show-Don't-Tell" approach).
- **Career Readiness:** Technical interview preparation, API design principles review.
- **Final Project Completion:** Final polish, deployment, and presentation of the Capstone Project (e.g., an E-commerce system or a Collaborative Dashboard).

Capstone Project Example & Core Competencies

6.1 Capstone Example: Collaborative Task Manager.

- **Frontend (React):** Create a single-page interface with drag-and-drop task organization (e.g., Kanban board).
- **Backend (Django):** Implement user and team authentication, and a REST API for task creation, assignment, and status updates.
- **Database (PostgreSQL):** Design complex relational schemas for users, teams, and tasks.
- **Deployment:** Continuous deployment setup (CI/CD) where changes are automatically pushed to production after testing.

6.2 Final Portfolio Requirements. The Lurnxo standard requires every student to have a public portfolio showcasing the following:

- **A Minimum of 3 Deployed Projects:** Including the responsive landing page, the advanced React app, and the full-stack Capstone.
- **Professional GitHub Profile:** Organized repositories with clear README files, licensing, and commit history.
- **Live Demo Links:** Each project must be live and accessible on the web.

Detailed Weekly Breakdown & Focus Areas

Weeks 1-8: Foundation & Frontend

W1: HTML Structure & Semantics.
W2: CSS Positioning & Box Model.
W3: Flexbox & Grid (Responsive Layouts).
W4: JavaScript Variables, Functions, Logic.
W5: Asynchronous JS (Promises/Fetch).
W6: DOM Manipulation & Events.
W7: React Components, State, Props.
W8: React Hooks ('useEffect', Custom Hooks).

Weeks 9-16: Full Stack & Deployment

W9: Advanced React State & Routing.
W10: Python Programming (OOP).
W11: Python Data Structures & Algorithms.
W12: Django Models, ORM, Migrations.
W13: Django REST Framework (API).
W14: SQL & PostgreSQL Querying.
W15: Git/GitHub & Cloud Deployment.
W16: AI Tools, Security, and Portfolio Finalization.

This syllabus is subject to minor adjustments to reflect the latest industry trends.